# Quantum Information Dynamics of the Source and the Placed Source

Lindblad Dynamical Scission, von Neumann Entropy Production, and Landauer-Bounded Quantum-Classical Transduction

**Formal Proof Companion**

Charles D. Schaper, Ph.D.
*Annals of the Chrysene Formalism*, Volume 1, Number 4
Article 96, November 2026

## 1 Formal scope and result

Lean 4.24.0 freshly compiles **35 named finite theorems**. Thirty-three theorems have an empty axiom dependency list; the corrected-twirl and collective-flip certificates use Lean's standard propositional extensionality axiom (`propext`). No `sorry`, user axiom or native-evaluation axiom is used. The executable source is core-only; finite decisions are checked by the Lean kernel.

This is a partial formal companion, not a formalization of the entire quantum-information manuscript. It does not prove general complete positivity, continuous-time Lindblad convergence, trace-norm estimates, Spohn's inequality, general von Neumann or Umegaki entropy identities, Landauer work bounds, channel capacities, real logarithmic thresholds, or chemistry. The 15-state and variable-length tests are performed separately in Python. The Lean rank-one jump and twirl certificates use four-dimensional finite models; agreement identities use the full 64-word six-site model.

## 2 Inherited listing and correction

The supplied formal-proof dossier claimed four completed proofs and printed an axiom-free audit. Extraction and execution with Lean 4.24.0 instead fail all four proofs. In particular, `rfl` does not prove conjunctions of unrelated equalities, and the trace proof leaves local definitions unexpanded. Its quotient dimensions $4^5$ and $4^6$ also do not describe the revised six-site registry model, whose dimensions are 32 and 64. The supplied text is retained as historical evidence, not a valid kernel certificate.

The new source certifies the actual agreement transform, finite dissipator coefficients and corrected twirl. It removes the unrelated buffer-amplification arithmetic and replaces incorrect dimensional attributions. A finite commuting projector by itself is not claimed to prove a general decoherence-free subsystem.

## 3 Definitions and coverage

**Finite jump identities.**

The integer-valued function `jump` is the doubled matrix coefficient of the dissipator of $|u\rangle\langle y|$. Exhaustive four-state theorems certify trace zero on every matrix unit, diagonal population coefficients and off-diagonal damping. The admitted-sector Boolean dephaser is zero within a sector and $-1$ across sectors in doubled units.

**Corrected twirl.**
The four-dimensional double-index twirl sends each unit $E_{ij}$ to $\delta_{ij} I/4$. A separate certificate gives zero trace on $E_{11}$ for the inherited defective one-index formula. The dimension and Kraus-count arithmetic is stated separately from channel identities.

**Agreement transform.**
The explicit encoding of all 64 six-bit words is in range and injective. Complementation of all site bits changes only the frame bit. Five parity-coordinate theorems certify nearest-neighbour $ZZ$ action on the registry coordinates. The partial-trace condition on every pair of encoded words is invariant under collective complementation.

**Reversal and noise.**
Five-bit reversal is an involution, with eight fixed words and twelve nontrivial pairs. Pointer-state phases cancel; a specific off-diagonal phase does not. These finite certificates agree with the manuscript's distinction between a protected subsystem under collective flip and general local noise.

**Arithmetic boundaries.**
Dimension, entropy-bit and rational-retention cross-products are checked as integer propositions. They do not formalize logarithms, thermodynamics, the absorbing-chain recursion or molecular probabilities.

# 4 Reproduction and audit

Run `python audit_lean.py` with Lean 4.24.0 on the path, or set the `LEAN` environment variable to its executable. The audit verifies the source SHA-256, compiles the file and requires the declared axiom report for every named theorem. The source and resulting `formal_audit.json` are included. No external mathematical library is needed.

# 5 Complete executable source

**Article 96 Lean source**

```
/- Article 96: finite kernel certificates, Lean 4.24 core only.
   This replaces the supplied non-compiling listing. No analytic claims
   or chemical realization are certified by these finite statements. -/
namespace Article96
set_option maxRecDepth 100000
set_option maxHeartbeats 0
set_option maxRecDepth 100000
-- Doubled dissipator coefficient for the rank-one jump |u><y|.
def delta (i j : Nat) : Int := if i == j then 1 else 0
def jump (u y i j a b : Nat) : Int :=
  2 * delta i u * delta j u * delta a y * delta b y -
  (delta i y + delta j y) * delta i a * delta j b
-- q labels the admitted sector; its doubled dephaser is -1 across sectors.
def sec (a b : Bool) : Int :=
  2 * (if a && b then 1 else 0) -
  (if a then 1 else 0) - (if b then 1 else 0)
def word (w : Nat) (i : Nat) : Nat := (w / 2^i) % 2
def reg (w : Nat) (i : Nat) : Nat := (word w i + word w (i+1)) % 2
def encode (w : Nat) : Nat :=
  word w 5 * 32 + reg w 4 * 16 + reg w 3 * 8 +
  reg w 2 * 4 + reg w 1 * 2 + reg w 0
-- One-index defective twirl diagonal numerator for matrix unit E_ii.
def badTrace (i : Nat) : Nat := if i == 0 then 32 else 0
-- Corrected double-index twirl: independent shift and output projector.
def twirlDiag (i j a : Nat) : Nat :=
  (List.range 4).foldl (fun n shift =>
    n + if Nat.xor i shift == a && Nat.xor j shift == a then 1 else 0) 0
-- Integer matrix-unit trace after global frame flip.
def pt (a b : Nat) : Bool := a / 32 == b / 32

theorem source_dimensions : 15 = 6 + 9 := by decide

theorem source_sinks : 7 + 8 = 15 := by decide

theorem transient_rate_scale : 2 * 1 = 2 := by decide

theorem sector_dephaser : forall a b : Bool, sec a b = (if a == b then 0 else
    -1) := by decide

theorem jump_trace : forall u y a b : Fin 4, ((List.range 4).map (fun i => jump
    u.val y.val i i a.val b.val)).sum = 0 := by decide

theorem jump_offdiag : forall u y i j : Fin 4, i != j -> jump u.val y.val i.val
    j.val i.val j.val = -(delta i.val y.val + delta j.val y.val) := by decide

theorem jump_diagonal : forall u y a i : Fin 4, jump u.val y.val i.val i.val a.
    val a.val = 2 * delta i.val u.val * delta a.val y.val - 2 * delta i.val y.
    val * delta i.val a.val := by decide

theorem inherited_twirl_not_tp : badTrace 1 = 0 := by decide

theorem corrected_twirl_units : forall i j a : Fin 4, twirlDiag i.val j.val a.
    val = (if i.val == j.val then 1 else 0) := by exact of_decide_eq_true rfl

theorem twirl_trace_scale : 32 * 1 = 32 := by decide
```

```
theorem kraus_count : 32 * 32 = 1024 := by decide

theorem registry_dimension : 2 ^ 5 = 32 := by decide

theorem site_dimension : 2 ^ 6 = 64 := by decide

theorem agreement_range : forall w : Fin 64, encode w.val < 64 := by decide

theorem agreement_injective : forall w v : Fin 64, encode w.val = encode v.val
    -> w = v := by decide

theorem agreement_collective_flip : forall w : Fin 64, encode (63-w.val) = Nat.
    xor (encode w.val) 32 := by decide

theorem partial_trace_collective : forall w v : Fin 64, pt (encode w.val) (
    encode v.val) = pt (encode (63-w.val)) (encode (63-v.val)) := by decide

theorem agreement_parity_0 : forall w : Fin 64, word (encode w.val) 0 = reg w.
    val 0 := by decide

theorem agreement_parity_1 : forall w : Fin 64, word (encode w.val) 1 = reg w.
    val 1 := by decide

theorem agreement_parity_2 : forall w : Fin 64, word (encode w.val) 2 = reg w.
    val 2 := by decide

theorem agreement_parity_3 : forall w : Fin 64, word (encode w.val) 3 = reg w.
    val 3 := by decide

theorem agreement_parity_4 : forall w : Fin 64, word (encode w.val) 4 = reg w.
    val 4 := by decide

def rev5 (w : Nat) : Nat := word w 0 * 16 + word w 1 * 8 + word w 2 * 4 + word w
     3 * 2 + word w 4

theorem reversal_involution : forall w : Fin 32, rev5 (rev5 w.val) = w.val := by
     decide

theorem palindrome_count : ((List.range 32).filter (fun w => rev5 w == w)).
    length = 8 := by decide

theorem nonpalindrome_pairs : ((List.range 32).filter (fun w => rev5 w != w)).
    length = 2 * 12 := by decide

theorem uniform_pure_entropy_bits : 5 - 0 = 5 := by decide

theorem uniform_pair_entropy_bits : 5 - 1 = 4 := by decide

theorem retention_numerator : 10000 * 10000 = 100000000 := by decide

theorem retention_denominator : 10001 * 10001 = 100020001 := by decide

theorem retention_below_one : 100000000 < 100020001 := by decide

theorem retention_above_9998 : 100000000 * 10000 > 9998 * 100020001 := by decide

theorem sink_coherence_zero_rate : (0 : Int) + 0 = 0 := by decide

```

```
theorem phase_noise_pointer : forall w : Fin 32, ((word w.val 0 + word w.val 0)
    % 2) = 0 := by decide

theorem phase_noise_offdiagonal : (word 0 0 + word 1 0) % 2 = 1 := by decide

theorem six_site_matrix_units : 64 * 64 = 4096 := by decide

end Article96
#print axioms Article96.source_dimensions
#print axioms Article96.source_sinks
#print axioms Article96.transient_rate_scale
#print axioms Article96.sector_dephaser
#print axioms Article96.jump_trace
#print axioms Article96.jump_offdiag
#print axioms Article96.jump_diagonal
#print axioms Article96.inherited_twirl_not_tp
#print axioms Article96.corrected_twirl_units
#print axioms Article96.twirl_trace_scale
#print axioms Article96.kraus_count
#print axioms Article96.registry_dimension
#print axioms Article96.site_dimension
#print axioms Article96.agreement_range
#print axioms Article96.agreement_injective
#print axioms Article96.agreement_collective_flip
#print axioms Article96.partial_trace_collective
#print axioms Article96.agreement_parity_0
#print axioms Article96.agreement_parity_1
#print axioms Article96.agreement_parity_2
#print axioms Article96.agreement_parity_3
#print axioms Article96.agreement_parity_4
#print axioms Article96.reversal_involution
#print axioms Article96.palindrome_count
#print axioms Article96.nonpalindrome_pairs
#print axioms Article96.uniform_pure_entropy_bits
#print axioms Article96.uniform_pair_entropy_bits
#print axioms Article96.retention_numerator
#print axioms Article96.retention_denominator
#print axioms Article96.retention_below_one
#print axioms Article96.retention_above_9998
#print axioms Article96.sink_coherence_zero_rate
#print axioms Article96.phase_noise_pointer
#print axioms Article96.phase_noise_offdiagonal
#print axioms Article96.six_site_matrix_units
```